Shkenca Kompjuterike 2

Blerim Zylfiu - blerim.zylfiu@ubt-uni.net ,
Lavdim Menxhiqi - lavdim.menxhiqi@ubt-uni.net

Inheritance - Trashëgimia

Trashëgimia është një shtyllë e rëndësishme e OOP (Programimi i Orientuar në Objekt). Është mekanizmi në java me të cilin një klasë lejohet të trashëgojë tiparet (fushat dhe metodat) e një klase tjetër.

Terminologji të rëndësishme:

Super Klasa: Klasa tiparet e së cilës trashëgohen njihet si super klasë (ose një klasë bazë apo një klasë prindërore).

Nënklasa: Klasa që trashëgon klasën tjetër njihet si nën klasë (ose një klasë e prejardhur, klasë e zgjatur ose klasë e fëmijëve). Nënklasa mund të shtojë fushat dhe metodat e veta përveç fushave dhe metodave të super klasës.

Ripërdorimi: Trashëgimia mbështet konceptin e "ripërdorimit", d.m.th. kur duam të krijojmë një klasë të re dhe tashmë ekziston një klasë që përfshin disa nga kodet që duam, ne mund ta nxjerrim klasën tonë të re nga klasa ekzistuese. Duke bërë këtë, ne jemi duke ripërdorur fushat dhe metodat e klasës ekzistuese.

Qëllimi i trashëgimisë

Ideja prapa trashëgimisë në Java është që ju të mund të krijoni klasa të reja të ndërtuara mbi klasat ekzistuese.

Kur trashëgoni nga një klasë ekzistuese, mund të ripërdorni metodat dhe fushat e klasës prind.

Për më tepër, ju gjithashtu mund të shtoni metoda dhe fusha të reja në klasën tuaj aktuale.

Sintaksa e Java Trashëgimisë

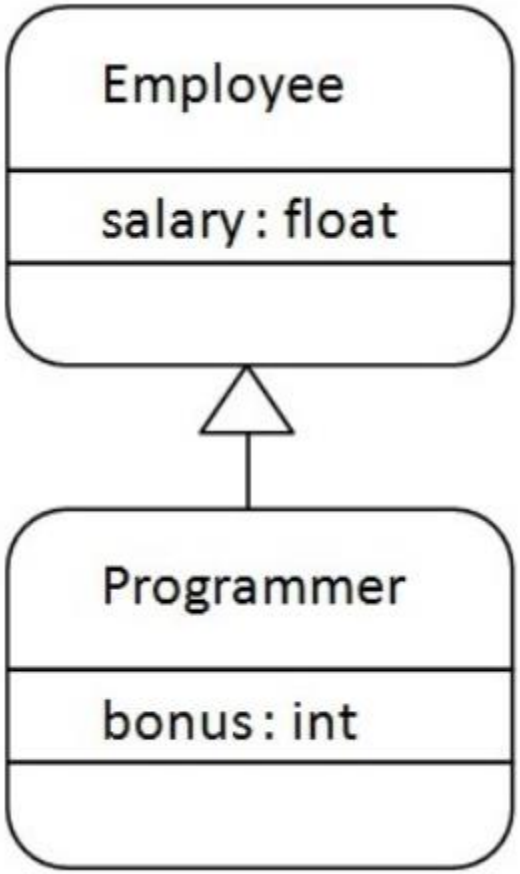
Sintaksa :

```
class Subclass-name extends Superclass-name
{
   //methods and fields
}
```

Fjala extends tregon që ju jeni duke bërë një klasë të re që buron nga një klasë ekzistuese. Kuptimi i "extends" është të rrisë funksionalitetin.

Në terminologjinë e Java, një klasë e cila trashëgohet quhet prind ose superklasë, dhe klasa e re quhet fëmijë ose nënklasë (Subclass).

Shembull



Siç tregohet në figurë, Programeri është nënklasa dhe Punonjësi është superklasa. Marrëdhënia midis dy klasave është Programeri **IS-A** Punonjës. Kjo do të thotë që Programeri është një lloj i Punonjësve.

```
public class Employee{
    float salary=4000.0f;
}
```

```
public class Programmer extends Employee{
    int bonus=1000;
    public static void main(String args[]){
        Programmer p=new Programmer();
        System.out.println("Programmer salary is:" +
                                                p.salary);
        System.out.println("Bonus of Programmer is:"+
                                                p.bonus);
    }
}
```

Output:

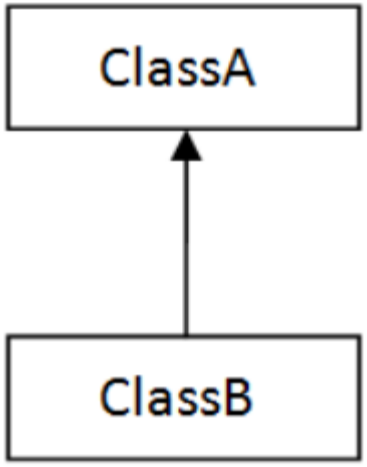
```
Programmer salary is:4000.0
Bonus of programmer is:1000
```

Në shembullin e mësipërm, objekti Programmer mund të ketë qasje në fushat e klasës vetanake, si dhe të klasës Employee, dmth., Ripërdorimi i kodit.

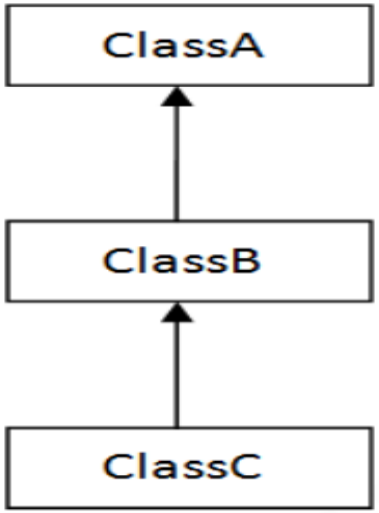
Llojet e trashëgimisë

Në bazë të klasës, mund të ekzistojnë tri lloje të trashëgimisë në java: të vetme (single), shumë nivele (multilevel) dhe hierarkike (hierarchical).

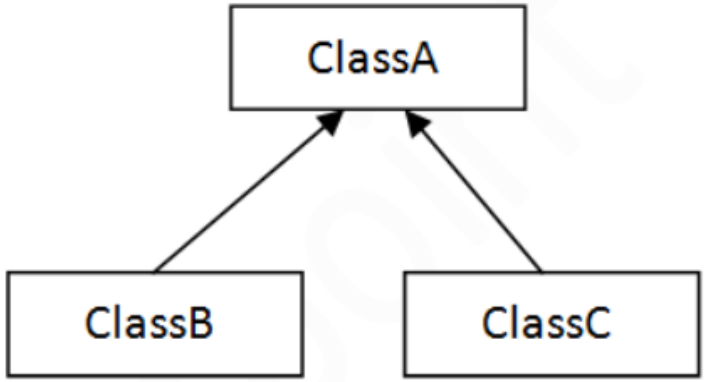
Në java, trashëgimia e shumëfishtë dhe hibride mbështetet vetëm përmes ndërfaqes (interface). Do të mësojmë më vonë për ndërfaqet (interface).



1) Single



2) Multilevel



3) Hierarchical

Shënim: Trashëgimia e shumëfishtë nuk mbështetet nga Java (përmes klasës).

Trashëgimia e shumëfishtë

```
// First Parent class
class Parent1
{
    void fun()
    {
        System.out.println("Parent1");
    }
}

// Second Parent Class
class Parent2
{
    void fun()
    {
        System.out.println("Parent2");
    }
}

// Error : Test is inheriting from multiple
// classes
class Test extends Parent1, Parent2
{
   public static void main(String args[])
   {
       Test t = new Test();
       t.fun();
   }
}
```

një klasë mund të trashëgojë veti të më shumë se një klase prind. Problemi ndodh kur ekzistojnë metoda me emërtim të njëjtë si në super klasë ashtu edhe në nënklasë. Me thirrjen e metodës, compiler nuk mund të përcaktojë se cila metodë e klasës do të thirret.

Output : Compile Error

Protected (modifier)

- Dukshmëria (Visibility) i variablave dhe metodave tregojnë se cilat variabla mund të trashëgohen e cilat jo
- Variablat dhe metodat me dukshmëri të deklaruar si ***public*** mund të trashëgohen, ndërsa ato me dukshmëri ***private*** nuk mund të trashëgohen
- Variablat e deklaruara si ***public*** nuk i përmbahen kërkesës për "mbërthim" (encapsulation)

Ekziston edhe forma e tretë e dukshmërisë që ndihmon gjatë "inheritance" e njohur si : ***protected***

Protected

- ***protected*** mundëson që variabla apo metoda të trashëgohet nga superklasa në nënklasë

- Dukshmëria e ***protected*** i përmbahet më tepër kërkesës për "mbërthim" se sa që ofron dukshmëria public

- Mirëpo "mbërthimi" në protected nuk ofrohet ashtu siç ofrohet përmes dukshmërisë ***private***

Metodat dhe variablat protected në UML Diagram mund të fillojnë me #.

Referenca super

- Konstruktorët edhe pse mund të kenë dukshmëri public, nuk mund të trashëgohen
- Pa marrë parasysh neve shpesh na nevojitet t'i qasemi konstruktorit të supërklasës në mënyrë që t'i inicilizojmë variablat e supërklasës
- Referenca ***super*** është fjalë e rezervuar, përdoret t'i referohemi superklasës dhe shpesh përdoret që të thirret konstruktori i superklasës

Referenca super

- Konstrukori i nënklasës është pjesa ku thirret konstrutori i superklasës
- Brenda konstruktorit lokal, nësë veç thirret konstruktori i superklasës, atëherë kjo duhet të jetë vija e parë e kodit, pra vija e parë e kodit në këtë rast i qaset referencës super
- Referenca ***super***, mund të përdoret të thirren variablat dhe metodat e supërklasës nësë variablat e superklasës fshihen apo metodat mbishkruhen